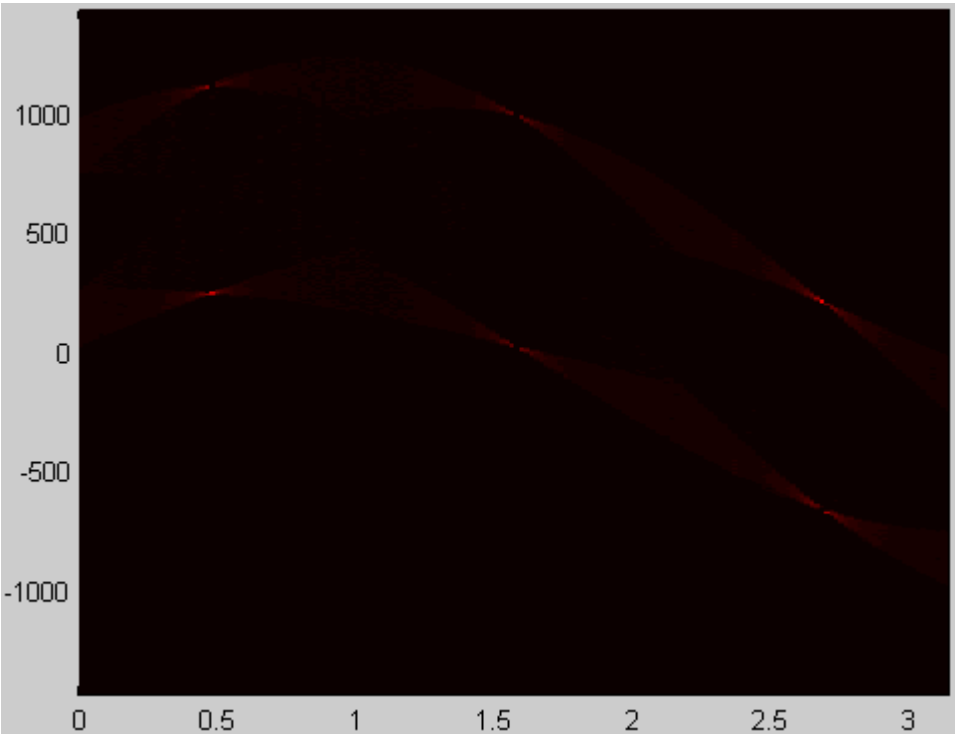
EECS 5330 Image Analysis and Computer Vision

Assignment 3

Hough Transform

By Edris Amin

November 8, 2011



In this project we were to write a MATLAB program to detect the angle of straight line edges of a polygon using the Hough transform. The image in figure 1a was used as the source polygon. My program was written to accept an image of any size as long as it was square. The image in figure 1b was the resulting edge image after the gradient operator was performed on the original.

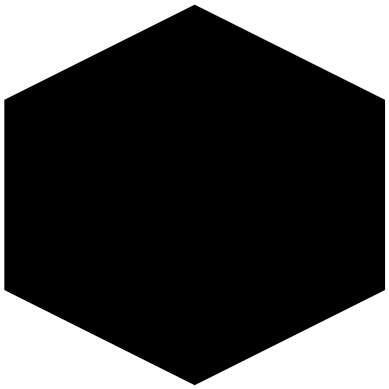


Figure 1a

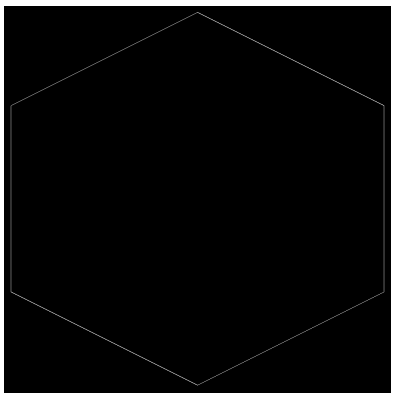
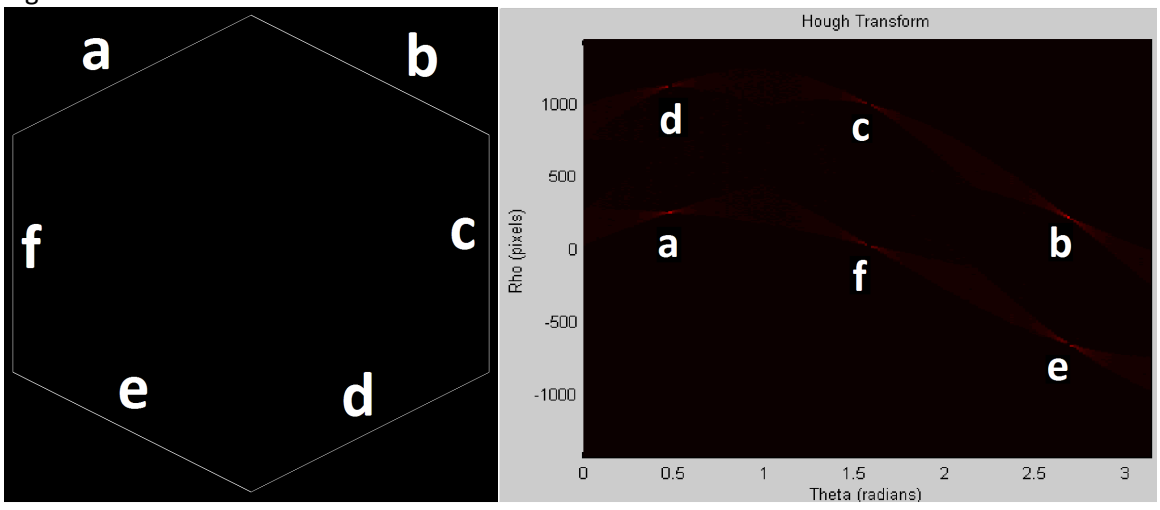


Figure 1b

The process for the hough transform followed the following algorithm found in *Digital Image Processing* by Gonzales.

"1. Compute the gradient of an image and threshold it to obtain a binary image
2. Specify subdivisions in the pTheta-plane
3. Examine the counts of the accumulator cells for high pixel concentrations.
4. Examine the relationship (principally for continuity) between pixels in chosen cell."

The image in figure 2 is the representation of the edges (straight lines) detected in the gradient image in figure 1b. The points of interest are the brighter points where many lines are intersecting. Those points represent a straight line corresponding to the angle on the horizontal axis and the distance from the origin on the vertical axis. This can be seen with comparing the labeled edges and the detected edges in figure 2.



Code

```
clear all;
poly = im2double(imread('poly.bmp', 'bmp')); %loading my polygon as double
for use in gradient sqrt() operation
[jz,iz]=size(poly); %get size of original image
polygrad = zeros(jz, iz); %create image of samesize for gradient image

for i=1:(iz-1) %perform gradient operation to detect edges of polygon
    for j=1:(jz-1)
              polygrad(i,j) = sqrt( (poly(i+1,j)-poly(i,j))^2 + (poly(i,j+1)-
      poly(i,j))^2 ); %gradient operator
    end
end

for s=1:jz
    polygrad(s,iz) = 0; %fill last column
end

for s=1:iz
    polygrad(jz,s) = 0; %fill last row
end

imwrite(polygrad, 'polygrad.bmp', 'BMP'); %save gradient image to disc

%hough transform process (taken from Gonzalez)
% 1. Compute the gradient of an image and threshold it to obtain a binary image
% 2. Specify subdivisions in the pTheta-plane
% 3. Examine the counts of the accumulator cells for high pixel concentrations.
% 4. Examine the relationship (principally for continuity) between pixels in chosen cell

% 1. gradient has been calculated     polygrad

% 2. rho-theta space
%maximum distance of pixel from image origin
rhoMax = sqrt(iz^2+jz^2); %~362
rho = (-rhoMax:1:rhoMax); %rho space {-362, 362}

%thetaStep = pi/180;%~0.01745
thetaStep = 0.01745;
theta = (0:thetaStep:pi);%theta space {0, 180} with 1deg step

hough = zeros(numel(rho), numel(theta));%hough transform space (rho x theta)

[jp, ip] = find(polygrad); %find edges in [rows, columns] of gradient edge
image

%initiate accumulator array
accumulator = zeros(numel(jp), numel(theta));

cosM = (0:jz)'*cos(theta);
sinM = (0:iz)'*sin(theta);
```

```
% 3. Examine the counts of the accumulator cells for high pixel
concentrations.
accumulator((1:numel(jp)),:) = cosM(jp,:) + sinM(ip,:);
for n=1:numel(theta)
    hough(:,n) = hist(accumulator(:,n), rho);
end

% 4. Examine the relationship (principally for continuity) between pixels in
chosen cell
pcolor(theta,rho,hough);
shading flat;
title('Hough Transform');
xlabel('Theta (radians)');
ylabel('Rho (pixels)');
colormap('hot');
```